United States

Department of the Interior

Geological Survey

AN ON-SITE SEISMIC DATA RECORDING SYSTEM

by

John Rogers

and

John Lahr

Open-File Report # 86-251

Menlo Park, California

1986

Table of Contents

Appendices

Acknowledgements

The University of California, Berkeley ASP system furnished much of the initial inspiration for the ELOG development and in particular, Ernie Major, one of the ASP developers was very helpful in some of the original conceptualizations. In addition Chris Stephens was heavily involved in the first array deployment in 1985. I wish also to thank Bob Page for his many important suggestions and insights.

1. INTRODUCTION

Many remote areas of the world are of great interest to seismologists, for example, Alaska within the United States. The acquisition of seismic data in these areas poses a difficult problem. The challenge has been met, at least to some extent, through the use of low-power VHF radios with multiple line-of-sight links to telemeter seismic data over long distances to the nearest town or communications facility. In rugged and remote terrain this has usually meant compromise in choosing sites, as well as high cost.

The Geological Survey embarked in 1984 on a project to develop a low-cost "smart" seismic station which could be used without VHF radio telemetry. The system had to be small, rugged, low-power and be able to record earthquake data on-site with accurate internal timing for an interval of at least 1 year. The system should also permit its data to be relayed via satellite when channels of sufficient baud rate become available.

The initial primary use for this instrument, an earthquake data logger or ELOG, is to study special regions out of range of the present VHF links set up in Alaska. Extending the current VHF radio links is not possible due to the high equipment costs involved in the links as well as in leasing the connecting commercial phone circuits. A possible secondary use of the ELOG is for rapid deployment in an aftershock investigation.

Because on-site recording capacity is limited, entire earthquake events cannot usually be recorded. Instead, the arrival time and certain key parameters are saved. An additional event buffer containing three seconds of pre- and nine seconds of post event data can also be saved if the investigation's duration is only several weeks or less. Depending on the level of seismicity, channel capacity and needed bandwidth for the data, the information could also be relayed by satellite.

The ELOG hardware is based on the commercially available RCA Microboard series of microcomputer products [1]. The CPU board handles all digital signal processing with a memory expansion board holding part of the program. Another RCA board handles tape I/O and a USGS designed board provides parallel to byte-serial time conversion for the 8-bit data bus as well as all analog signal conditioning. An 8-bit analog-to-digital (A-D) converter board digitizes the single channel of incoming data. Communication with the ELOG is via an RS232 port on the CPU board. In the field a portable terminal is used for ELOG setup. The ELOG system block diagram is shown in Figure 1.

The RCA microcomputer board is based on the CDP1805CE microprocessor [2, 3] which has been set up to run FORTH [4], a high-level computer language. Although similar in some respects to BASIC, FORTH is faster and more compact than BASIC, features that are of prime concern for small systems like the ELOG. Although assembly language is faster than FORTH it is not well suited to easy program modification, a necessary feature in a dynamic research situation. The code (4k bytes) is written in a combination of FORTH and assembly which can be called from FORTH.

Timing is provided by an Omega Radio synchronized clock which is accurate to about 20msec. Omega is a world-wide system run by the United States Coast Guard primarily for navigation. Its transmissions, however, are very

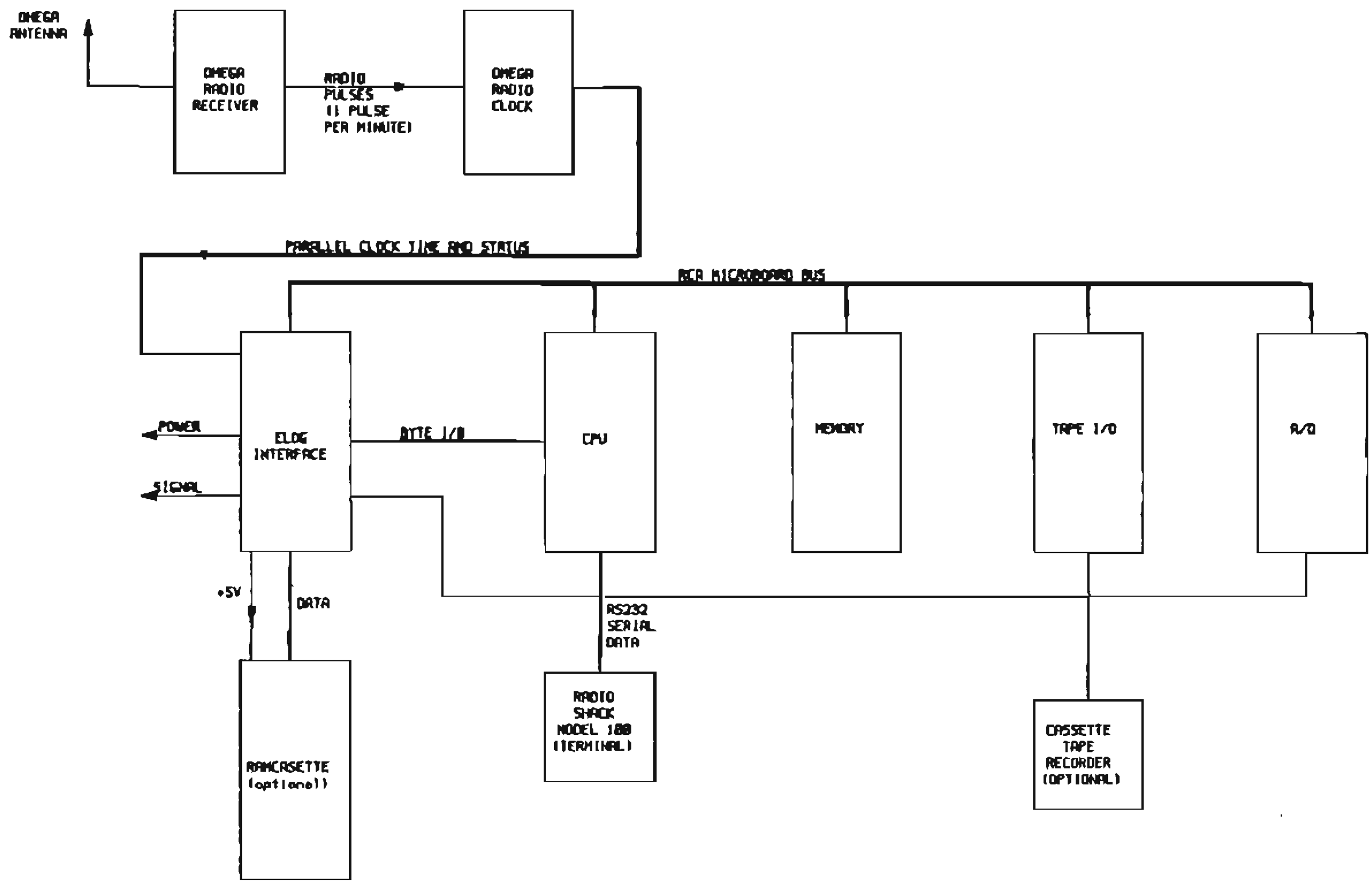


FIGURE 1 ~ ELOG SYSTEM BLOCK DIAGRAM

precisely timed which allows the synchronization of a clock from the radio pulses. The parallel time output of the clock is input to the ELOG and read whenever an event is detected. The radio/clock combination is a commercially available product [10].

The data are stored on magnetic tape cassettes or solid-state mass storage. Without waveform data, up to 2000 events can be stored on a single cassette tape or 3200 in a prototype solid-state mass storage. The later option could be expanded in the printed circuit board version from a minimum of 512k-bytes to 2M-bytes. For either recording configuration, the ELOG will run for an entire year on four 1000 ampere-hr batteries. During this time the program monitors incoming data for what might be an event. Parameters for events that pass a screening to eliminate low frequency arrivals and noise bursts are collected in mass storage.

The heart of the program is a sampling loop which continually updates a long term average (LTA) and a short term average (STA) of the data. Each time through the loop these averages are compared to see if the STA exceeds the LTA by a factor of three. If so the 12 second event buffer (starting when the STA is exceeded and lasting 9 seconds) is tested for frequency content, energy and event impulsiveness. If the tests are passed, the event parameters are written to a temporary buffer which holds 19 events. After the 19th event the buffer is written out to mass storage.

The stored data is detailed below with the number of hexidecimal characters in the output in parenthesis

1) Clock Status(1)-indicates whether the clock is properly synchronized to Omega

2) Detection Time(11)-The instant when the STA reaches or exceeds the LTA by a factor of three, expressed in terms of Julian day, hour, minute, second and hundredth of a second

3) PCTR (2)- Onset Time-The instant before detection time when the STA reaches or exceeds twice the LTA expressed as the number of samples prior to the detection time

4) IMOT (4)- Onset Sample Value-The value of the data point at the onset time. This also gives the polarity of the first motion.

5) IMAX (2)- Maximum Sample-The absolute value of the largest amplitude during the first half-cycle.

6) TMXS (2)- Time Until First Zero Crossing-The number of samples to the first change of polarity measured from the onset time

7) N3LT (4)- Energy Level-The number of times during the first nine seconds of the event that the STA drop below twice the LTA. This provides information on the energy duration of the event.

8) LTA(2)- The LTA value before the event. This gives a measure of the background noise.

9) ECTR(4)- Number of triggers.

10) NZRO(4)- Number of zero crossing during the first 9 seconds of event.

The other sections of this report will examine the hardware and software in more detail. All references to "screen numbers" refer to the RAM cassette version of the FORTH program (Appendix 3). Screen numbers are FORTH's way of dividing up a program into smaller units. The other program versions are Cassette Tape (Appendix 4) and RAM cassette with waveform storage which is currently under development. The basic elements of all three programs are very similar.

2. PROGRAM ORGANIZATION

The ELOG program is organized into three conceptual parts, only one of which will be executing at any given time. The central routine is the sampling loop (screen 25) which digitizes incoming data and places the values in a circular buffer. The STA and LTA are then updated (screens 15, 16) and compared to see if a threshold has been exceeded. At this point the sampling loop is either executed again with a new sample if the threshold test failed or program execution moves to the verify phase if the threshold was exceeded.

In the verify phase events are screened to minimize the recording of noise. The screening involves running three tests on the data in the buffer to determine which events to save. Amplitude and frequency content must match the type of earthquake being searched for and an emergence test screens out distant events. If these three tests are passed then the final recording phase is entered at which time the data is written to a RAM parameter buffer which is ultimately written to mass storage.

The ELOG software is written in FORTH, a computer language which finds major application in dedicated microcomputer-based control systems. FORTH is a threaded, interpretive, extensible language [5, 6, 7]. The term "threaded" has to do with the way FORTH code is compiled. In traditional compiled languages source code is translated into machine language as a unit and can only be executed as a unit. In FORTH each "word" is compiled separately and can be executed either by itself or with other FORTH words. When FORTH words are executed individually, operation is similar to other interpretive languages such as BASIC. When many words are combined and executed as a whole, operation is similar to the FORTRAN subroutine. Defining new words from already defined words or from assembly language gives FORTH its extensibility.

FORTH uses two stacks for inter-word communication and math. The stack orientation of FORTH makes the use of reverse polish operations necessary even though code becomes harder to read. All math is done in fixed point as FORTH does not support floating point.

Tape I/O and the terminal interface is accomplished by calls to a utility ROM(UT62) supplied by RCA. This ROM contains the usual utility routines for writing to tape, reading from tape, examining memory, etc. In addition, one UT62 routine automatically sets the ELOG baud rate to match that of the terminal (up to 1200 baud).

2.1 Data Sampling

The sampling routine is interrupt-driven by a counter internal to the 1805 CPU. The counter is preset to an initial value by the word ADPG (screen 2) which controls the sampling rate. Each time the counter goes through zero the CPU is interrupted and the data sampled. In the case where a higher sampling rate is desired, the counter is pre-set to a lower value. In practice the sampling rate is limited to 100 Hz or less by the CPU processing speed.

The sampling loop first selects the A-D by outputting a code 30H(hex) which conforms to the RCA two-level I/O addressing scheme [8]. This selection is necessary since there are several I/O devices connected to the CPU. The A-D is hard-wired to be selected by the 30H code. Then two 00H codes are output next to program the single channel mode and data channel zero. These two steps configure the A-D to digitize only on channel zero of the eight-channel A-D. Sampling is started as soon as the interrupt handler address is loaded(IADR) and the counter interrupt is enabled(CTEN). A semaphore(?RDY) is used to halt sampling loop execution until the next sample is ready. The interrupt handler code is presented in Appendix 1, while Screens 2 and 15 contain the above words.

Once the new sample is stored in memory, the word NXT converts the 8-bit offset binary code produced by the A-D to a 2's complement 16-bit code and stores the value in the variable YNEW. This value is placed in the circular buffer by the word PUTY. Then the STA and LTA are updated by the words SAVG and LAVG respectively. In order to increase processing speed SAVG and LAVG use divisions and absolute value operators written as assembly language FORTH "code" words. Since FORTH does not support floating point math, the code word divisions also put the remainder on the stack. Use of the remainder by SAVG and LAVG gives a more continuous LTA and STA.

Finally after the LTA and STA have been updated, their values are compared by the word ?EVT. ?EVT checks each time around the loop to see if the STA is three or more time the LTA. If this occurs, then the main sampling loop is temporarily exited and the verify section of the program is entered.

2.2 Verification of Events

Because memory is a limited resource the verification words attempt to prevent recording events that are not earthquakes. Before any tests are made however, the word RDTM (screen 2) reads the clock and stores the time in memory for later recording if the event is deemed real. Then the word TAIL collects another nine seconds of post-event data in the data buffer. TAIL (screen 22) uses a similar sampling loop to the one described above with two differences: the LTA and STA are not updated and the 1805 counter is re-initialized by ADPG to a large value to limit sampling to the 100 Hz rate.

Now that the above two time critical tasks are done, verification starts. First the energy of the wave is checked by AMPL. AMPL counts the number of times that the STA is less than two times the LTA. Then FREQ counts the number of zero crossings and SRCH finds the onset time. SRCH searches back in time for the point where the STA falls to two or less times the LTA. (Refer to screen 21-23). This point will tend to be early as the STA is an average and will not immediately follow the data. To compensate for this, the search is done again using a shorter averaging time (FAVG) and going forward in time to find the start of the event. Thus SRCH first looks back in time through the buffer for the approximate onset, which tends to be early, and then

forward in time for the true onset sample. This procedure prevents noise spikes just prior to the onset from being picked. Table 1 gives the parameters used in the search process as well as other ELOG parameters. When SRCH finishes executing, it produces a variable value which is a measure of the impulsiveness of the event. For the final recording phase of the program to be entered, the variables values produced by AMPL, FREQ, and SRCH must be within the bounds given in Table 1. Thus an event must maintain a certain energy level*, have at least a minimum number of zero crossings, and not be too emergent. Parameters for events that pass these tests are recorded.

2.3 Event Recording

Each time an event passes the three tests it is recorded in the parameter buffer. The parameter buffer holds 19 events which are written to mass storage as a group. Mass storage is either a low-cost cassette tape recorder or a solid-state memory array.

The cassette tape control software uses features built into MB4TH. SAVE-BUFFERS causes the entire parameter buffer contents to be written to tape. The variable USE which contains the number of the next buffer to be used is incremented and an update bit in the buffer is set.

With the RAMcassette, a memory location counter is used to address the memory array. CTRH and CTRL increment the counter, ENB and DISB activate and deactivate the RAMs, MEMRD and MEMWR select the read or write mode, and E->R and R->E do the reading or writing through the CDP1851CE programmable I/O port [9]. This port, like the A-D, must first be selected and programmed for the desired configuration. For example, when it is desired to write a byte to the RAMcassette, one port internal to the 1851 must be set up as an output port. These words are located in screens 3-6.

For either recording medium the parameters stored are the same. First the clock time which is available in parallel format is converted to byte-serial by the word RDTM. Each execution of RDTM initializes the 1851 as half input and half output. The output half sends control bytes to the time buffering electronics to control the order of reading through the input half of the port. Then the word WRTM moves the time bytes to the parameter buffer if the screening tests described above are passed. A compete memory map which shows the location of all ELOG memory areas is given in Figure 2.

The word FRST uses the onset time found by SRCH in the verification phase to find the direction of first motion(IMOT) and the maximum value sample during the first half cycle of the event(IMAX). SRCH also counts the number of sample points from the trigger time to the instant when the STA is two times the LTA. This value is stored in the variable PCTR. PCTR is used as an index to the circular buffer for the onset instant and is written to the parameter buffer by the word WRVR. WRBY writes the bytes IMOT and IMAX to the buffer. FRST then counts the number of samples to the first zero crossing from the onset and stores the value in TMXS which is also written to the buffer. Then, the number of times the STA falls below twice the LTA during the event (N3LT)

* If this level varies with the rate at which events are recorded, an automatic energy threshold adjustment can be achieved. This allows a recording rate which will just fill the available mass storage without having to re-visit the site to adjust this parameter. The FORTH words that implement this function are given in Appendix 6.

TABLE 1 - ELOG SETUP PARAMETERS

SPS, Sampling Rate, Samples Per Second	100
Energy Test, N3LT <	300
WIND, window on event after trigger	9sec
Zero Crossing Test, NZRO >	45
BOT, lowest address of data buffer	6000H
LTRG, onset threshold	2
NBUF, buffer length	2400 (bytes)
Emergence Test, < sample times	100 (1 sec)
Trigger threshold	3
LTA averaging time(LAVG)	4096
STA averaging time(SAVG)	16
FAVG fast averaging time	4

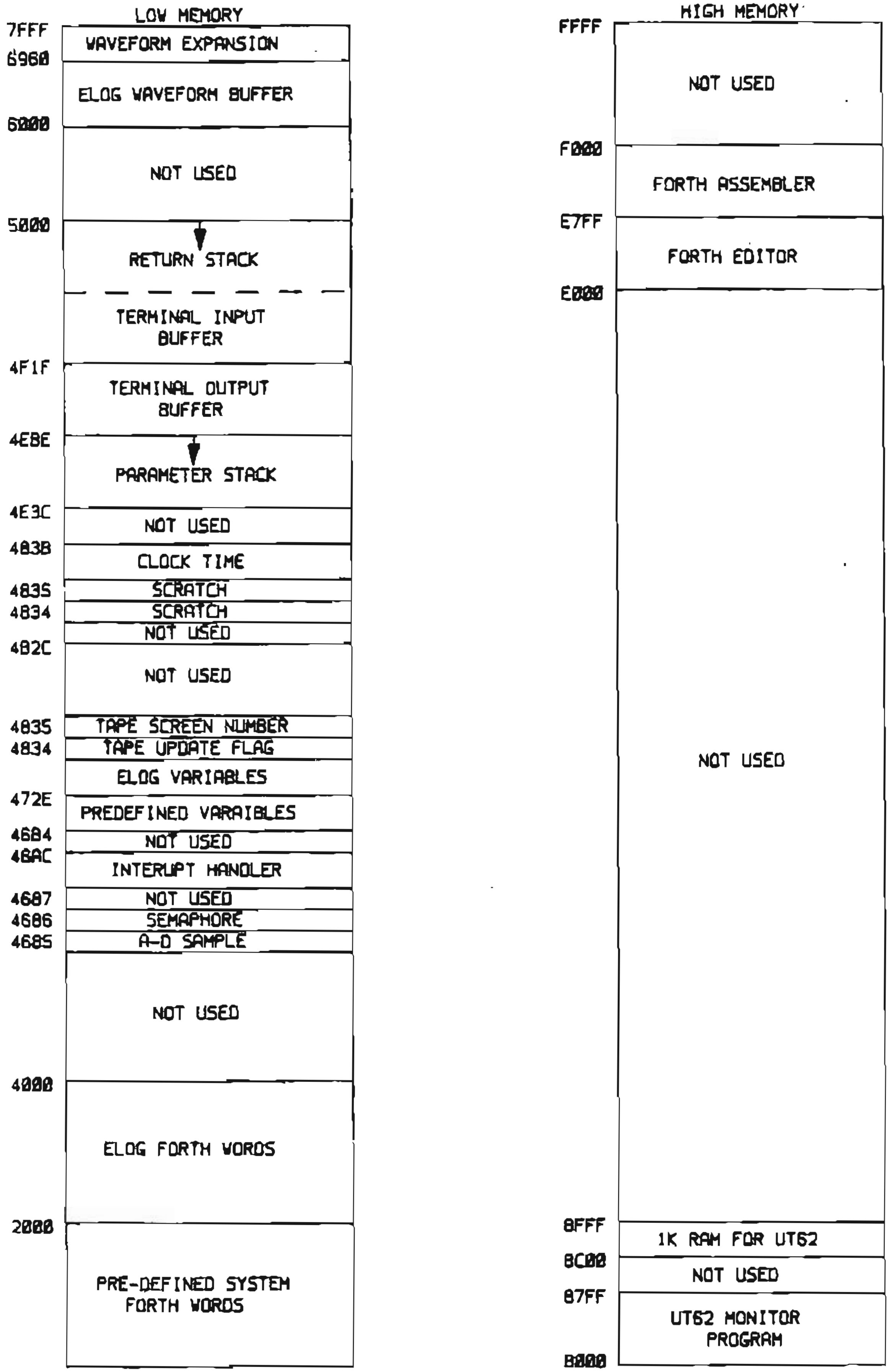


FIGURE 2 - MEMORY ALLOCATION MAP

is stored along with the LTA. Finally the number of triggers (ECTR) is stored and if the RAM cassette is used with waveform storage the index to the trigger time, YCTR, is also stored.

Each event record written to tape or RAM cassette storage contains 36 bytes. The format used is presented in Table 2 and the ELOG programs for the RAM Cassette and magnetic tape are in Appendices 3 and 4. Note that if the waveform is also stored then an extra 4 bytes of data are needed to indicate the trigger sample in the buffer relative to the buffer beginning, BOT.

3. SYSTEM HARDWARE

ELOG hardware consists of a card cage for the five printed circuit cards, an Omega Radio receiver, a clock, and either a cassette tape recorder or a RAMcassette. Figure 1 shows a block diagram of the system. As can be seen, the radio produces synchonizing pulses which keep the clock on time after it has been initially set. The parallel clock time is then available through the interface card to the CPU. The interface card also has amplifier and power converter sections. Program memory is divided between the CPU, Tape I/O and Memory expansion cards. A discussion of the various components follows.

3.1 CPU Card

The CPU card uses a CMOS 8-bit CDP1805CE microprocessor running from a 2MHz crystal. Instructions need an average of 16 clock cycles for execution resulting in a processing rate of 125,000 instructions per second. The card also has a serial RS232 port which is used for terminal communication at up to 1200 baud while a parallel programmable port is provided for byte input/output. The port is programmed via software instructions which determine whether the sections are to be used for input, output, or bi-directional data flow. The ELOG only uses the first two options. The card memory section has four 27C16 EPROMs which contain the FORTH compiler and initial dictionary, as well as four kilobytes of RAM. The external interrupt and polling features are not used. A wiring diagram of the RS232 cable is given in Appendix 5.

3.2 A-D Converter

The A-D converter card converts bipolar analog data to offset binary 8-bit samples. The RCA scheme for addressing I/O modules necessitates having each module respond to a "group select code". The code recognized by the A-D during an OUTPUT 1 instruction is 30H which is put on the bus as data.

After selection the A-D must be programmed for fixed digitizing from channel 0, the input channel. This is similar to the group select described above, except that 0H is used for data and two other output instructions are used for the programming.

After all these preliminaries, the A-D is ready to digitize its first sample. A "INPUT 3" instruction initiates the conversion. The code (in machine language) which handles the A-D, semaphore (discussed in Sec.2.1), and 1805 counter interrupt disabling is loaded by the word ILD (screen 19) and is executed once for each sample.

The A-D produces offset binary code which differs from 2's complement by the most significant bit. For 8-bit numbers, if 80H is subtracted from the offset binary number, a 2's complement number results. This number covers the range of input voltages from -/+ 2.5 volts. An expression which relates input voltage to counts is:

TABLE 2 - ELOG DATA FORMAT

FIELD		DATA DESCRIPTION
WAVEFORM	NO WAVEFORM	
1	1	clock status, 0=OK
2-12	2-12	BCD time-julian day, hr, min, sec, 1/10, 1/100 sec
13,14	13,14	(PCTR) sample time offset from trigger to onset
15-18	15-18	(IMOT) polarity-sample value at onset
19,20	19,20	(IMAX) maximum sample during first half cycle
21,22	21, 22	(TMXS) sample times from onset to first zero crossing
23-26	23-26	(N3LT) number of times STA falls below twice LTA
27,28	27,28	(LTA) long term average
29-32	29-32	(ECTR) number of triggers
33-36	33-36	(NZRO) number of zero crossing during wind
37-40	*	(ENTH) energy threshold
41-44	*	(YCTR) index to trigger point in buffer

Note: If waveform storage is selected then an additional 1200 bytes of waveform data follows immediately after the header. The header information is also displayed on a terminal if one is connected during the event.

input voltage (volts) = counts * 0.019532125

where counts is a hexadecimal number and must be converted to base 10 before multiplication.

3.3 Memory Expansion Board
The 1805 CPU can address up to 64k-bytes of memory of which 8k-bytes of ROM are on the CPU board along with 4k-bytes of RAM. Another 2k-bytes of ROM and 1k-byte of RAM are on the Tape I/O board. This leaves about 50k-bytes open for expansion of ELOG specific FORTH words. Due to the compactness of FORTH, only 4k-bytes are needed for the entire ELOG program which is in two 27C16 2k-byte by 8-bit CMOS EPROMs.

3.4 Tape I/O Board
The Tape I/O board serves the purpose of providing additional memory sockets and interfacing with the cassette tape recorders. The memory held on this board is the utility ROM(RCA UT62) and the FORTH editor and assembler. A CPU reset switch is also located on this board. When switched between reset and run, a signal is generated which causes the CPU to execute the utility program (UT62).

3.5 Interface Board
The Interface Board provides regulated power for the entire system, buffers the parallel time input from the clock and conditions the sensory input. Figure 3 displays the electronic schematic (in two sections) for this circuit.

3.51 Power Regulation
One LM309 3-pin voltage regulator provides +5 volts to power the system electronics except for the clock and radio which have their own internal batteries. The regulator is capable of supplying one ampere of which 500mA is available for external use. Another voltage converter(ICL7660) produces negative 5 volts for use by the signal conditioning section.

3.52 Signal Conditioning and Amplification
Signal conditioning consists of amplification to match the dynamic range of the A-D converter and filtering to prevent signal aliasing. The input signal from the L4 vertical geophone is applied to a low noise differential amplifier composed of a matched pair of transistors. The input impedance produces a critical damping factor of 0.7 with the gain fixed at 21.5. A potentiometer allows zeroing out of any offset. In the circuit shown in Figure 3, a cutoff frequency of 30Hz results. This cutoff is mainly influenced by the filtering action of C8/R10 and C7/R6. Amplifier U21 gives solid-state switchable gain control from a minimum of 340 to 44,000 in increments of 6dB. The maximum allowable signal coming out of the amplifier is limited to +/-2.5 V by the A-D.

3.53 Time Buffering
The internal batteries of the clock are 3.6 volt Lithium so clock logic levels need to be converted to 5-volt CMOS. This is accomplished by an array of RCA CD40109BE CMOS-CMOS level converters with three state outputs. Each CD40109 converts 4 bits of clock data which is input to the CDP1851CE I/O port when the CD40109 output drivers are activated by a high on the enable lines. The high level is produced by the word RDTM which sequentially reads two nibbles of data into the port starting at 1/100s seconds. The sequential reading is implemented by decoding the output bytes produced by RDTM with two 4028 decimal decoders. (The decoders are also used to control the RAMcassette by inputting different control bytes to the decoders.)

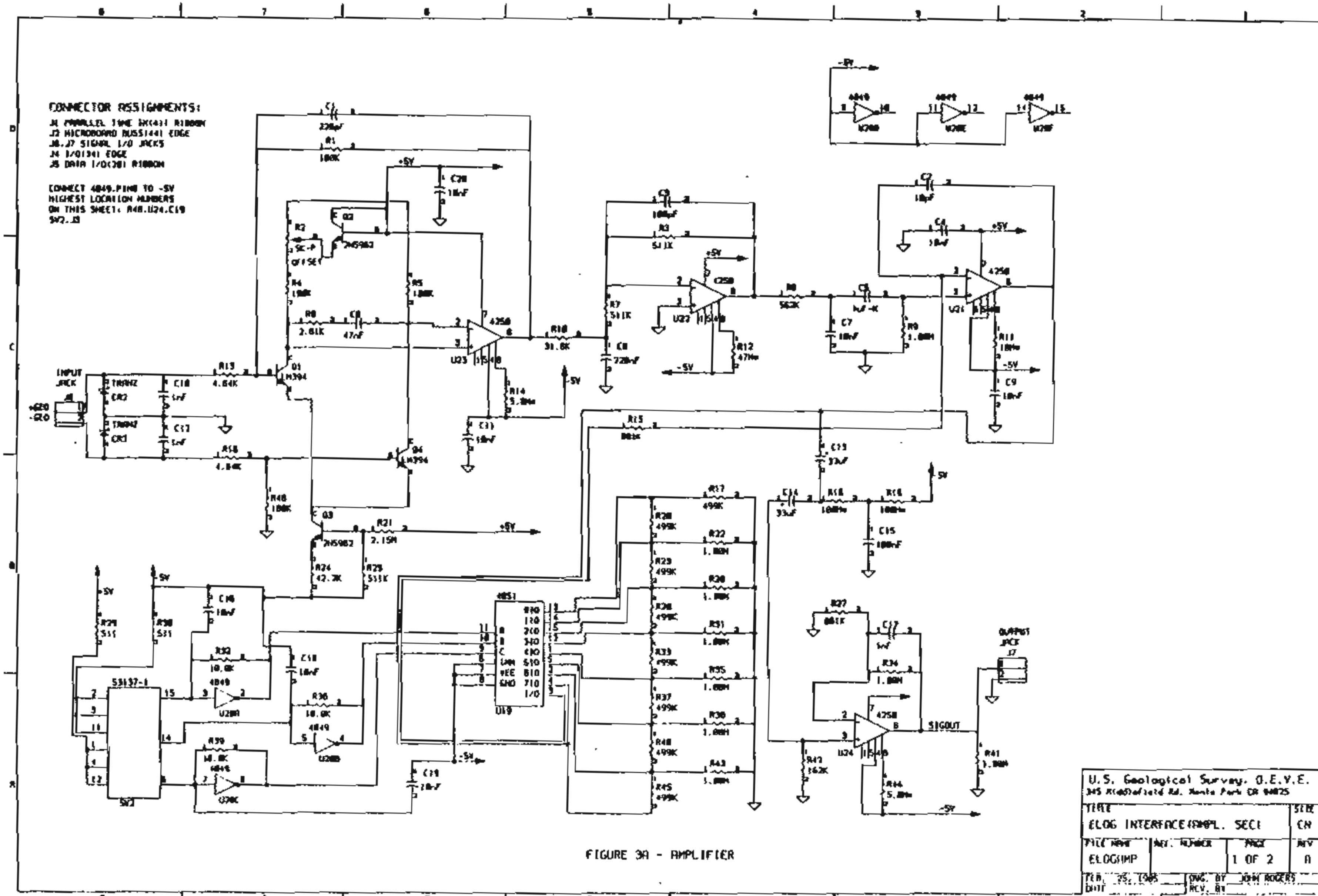


FIGURE 3A - AMPLIFIER

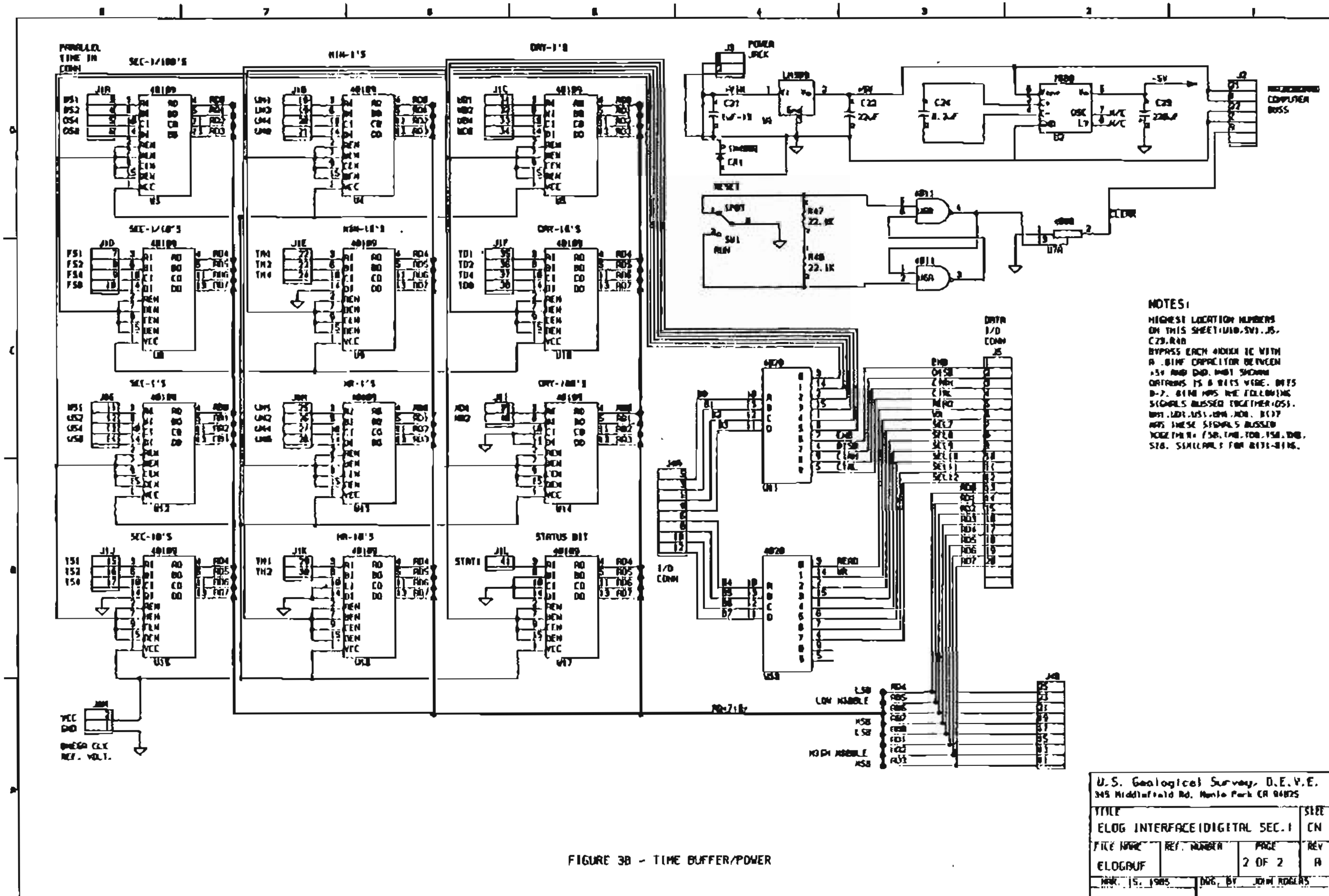


FIGURE 3B - TIME BUFFER/POWER

The wiring diagram for the clock-interface card cable is given in Appendix 5.

3.6 Mass Storage

The contents of the parameter buffer are dumped to the mass storage device every 19 events. Either a low cost cassette tape recorder, or RAMcassette, may be used. The RAMcassette is several times more expensive than the cassette but can operate reliably over a larger temperature range and hold more data.

The RAMcassette consists of an array of 8k-byte by 8-bit CMOS RAM chips which all share the same address lines. A chip select signal determines which RAM from the array is active. This signal is produced by a CD4516 CMOS 1 of 16 decoder. The circuit diagram of the 128k-byte version is shown in Figure 4.

The operation of the circuit is fairly straight forward. Five hexadecimal switches pre-load an address into the 17-bit counter chain U27-U31 when the reset button is pushed. This address determines where (in memory) array read/write operations will start. Bits 0 through 12 of the counters are used to address each RAM memory location while bits 13 through 16 are input to the decoder.

The counter is software controlled by the words CTRH and CTRL. Each execution of this pair of words advances the address by one count. The array is activated by the ENB word which removes the inhibit level from the decoder. The read or write mode is selected by the words MEMRD and MEMWR. All these signals first go through flip-flops before they are applied to any other circuit.

Playback of the RAMcassette data is handled by the word RDRECS which takes the number of records desired to be read off the stack. The data is outputted through the RS232 port where it can be transferred to floppy disk by a suitable terminal program running on a portable personal computer.

3.7 REAL TIME CLOCK

Real time is supplied to the ELOG through the interface card by an Omega Radio synchronized clock. Omega is a world-wide, low frequency, precisely timed radio signal whose six transmitters put out a repetitious code. Decoding is done by the Omega Radio receiver which is tuned to the characteristic frequency of the nearest Omega transmitter. The code is used to synchonize an oscillator inside the receiver so that the minute pulse output of the radio is always within 20msec of its theoretical value. This theoretical value is calculated by adding the offset between Omega (which uses Atomic Time), the propagation delay due to path length between the transmitter and receiver, and the internal receiver delay.

The delayed pulse output of the radio receiver is input to a clock whose time runs ahead of the radio pulses by the fixed amount calculated above. This offset advance is programmed by three BCD switches. Once set correctly, the delay in the radio pulses is cancelled by the advance in the clock so that the clock runs within 20msec of real time. If the receiver is unable to decode the incoming code, the pulse output's duty cycle is changed thereby informing the clock of a possible timing error. In this case the receiver oscillutor then free-runs on its own oscillator and the clock status bit is set to 1.

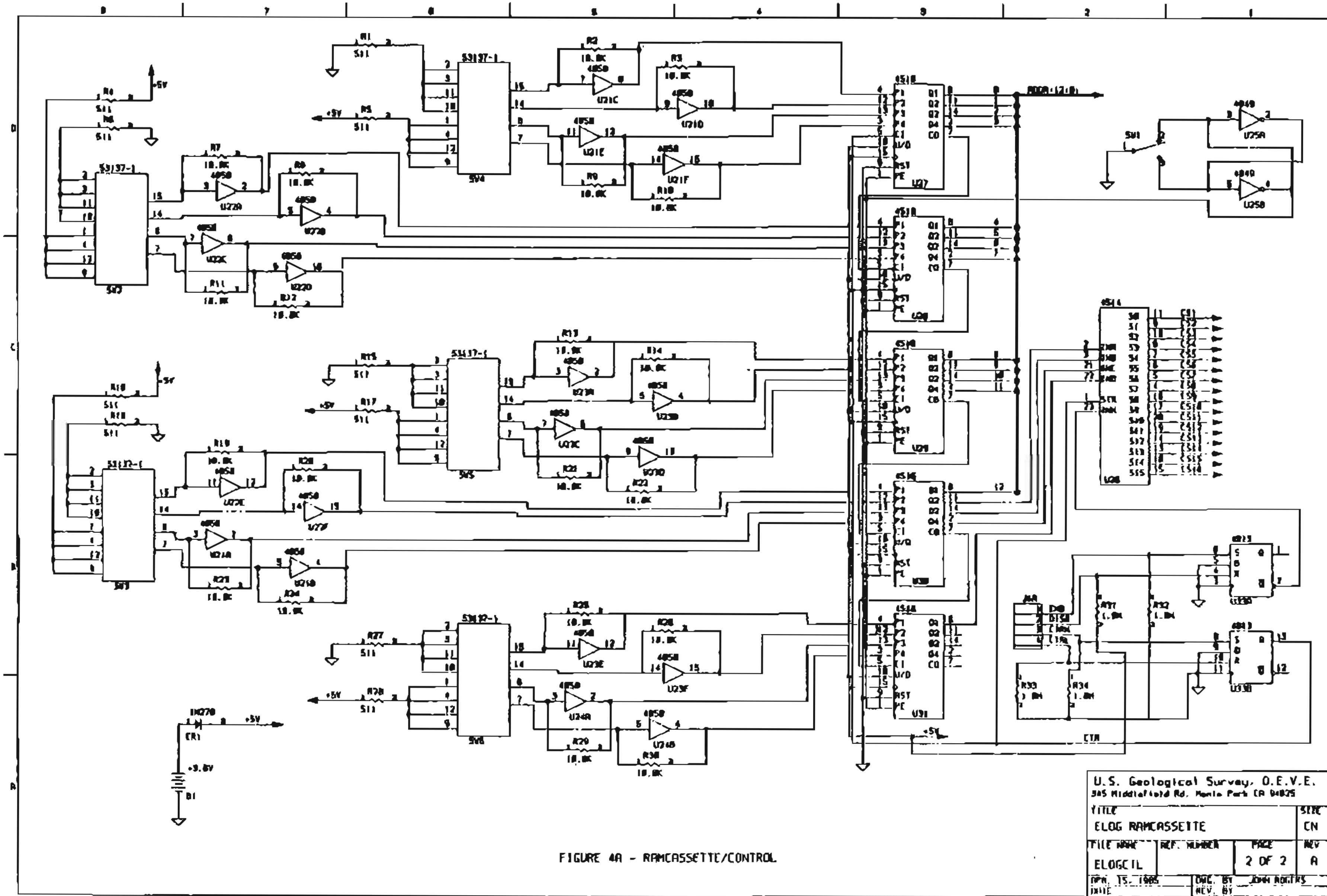


FIGURE 4A - RAMCASSETTE/CONTROL

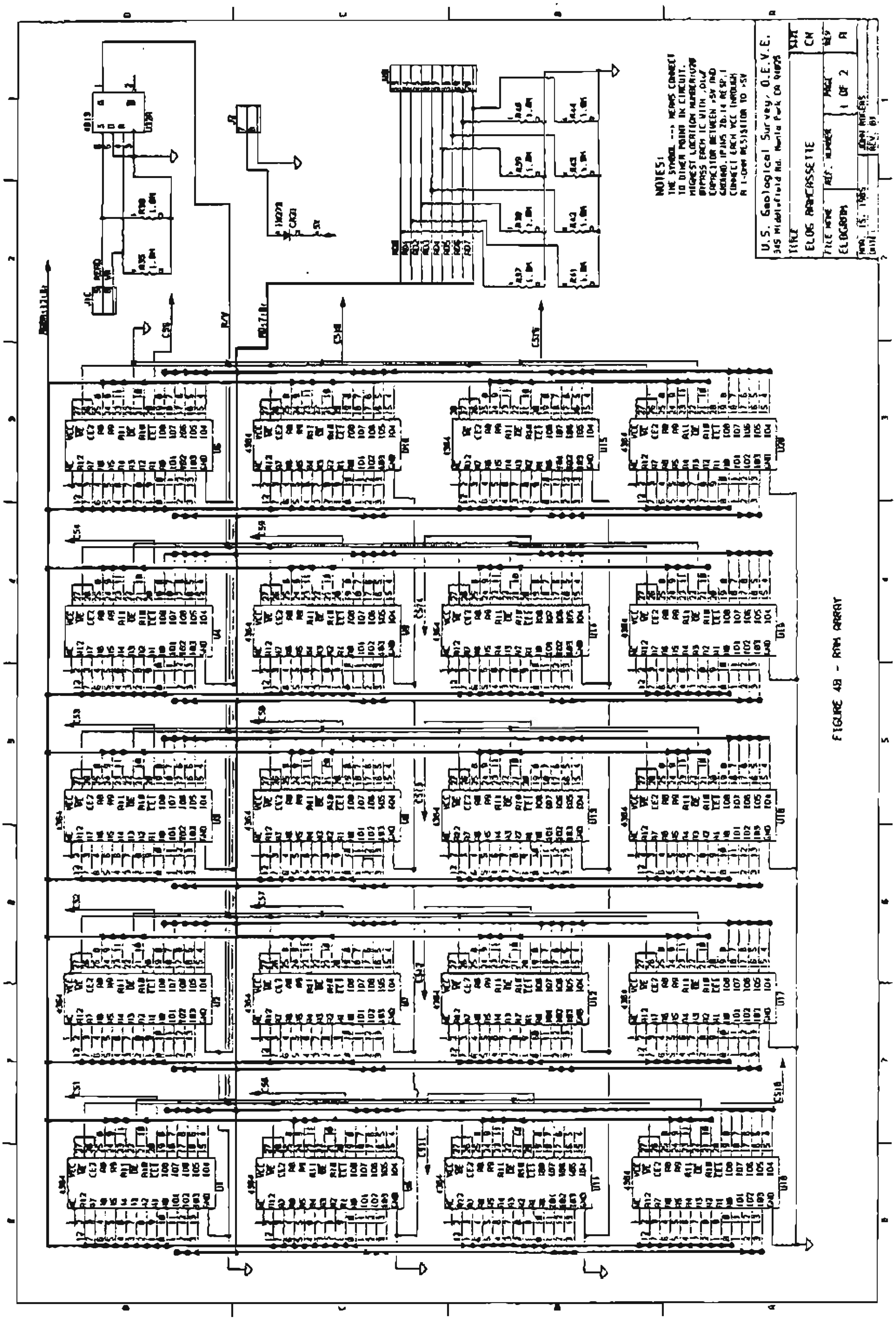


FIGURE 4B - RAM ARRAY

4.0 OPERATIONS

Conducting any successful field experiment involves the correct execution of many seemingly trivial tasks. Details for an ELOG deployment include selection of batteries and Omega radio receivers, the setting of the clock, burying the sensor and starting the ELOG program. The information below is intended to provide a guide for the deployment of an array of ELOGs. A wiring diagram is given in Figure 5 and the specifications in Appendix 1.

4.1 Battery Considerations

Although there are many different types of batteries available, the 6-volt rechargeable lead-acid, or the 2.5-volt non-rechargeable alkaline gelled electrolyte battery, probably are the most cost-effective choices for the ELOG power source. Lead-acid batteries are more suitable for deployments of several weeks, while alkaline batteries can run an ELOG for one year but cannot be reused.

The battery capacity needed to run an experiment depends on the length of the experiment and the temperature. Capacity is rated in ampere-hours or the number of hours of battery life at one ampere. Since the ELOG draws less than 0.1 ampere, the expected battery life is ten times the amp-hr rating, expressed in hours. Of this only 80 percent should be used and this number must be further reduced if operation below freezing is needed. The manufacturer's data sheet will contain the information needed to make an economical choice.

4.2 Setting the Clock

Accurate and reliable timing depends on a high signal-to-noise ratio of the Omega Radio signal. Thus the nearest Omega transmitter should be selected as the frequency of the Omega radio receiver. The antenna needs to be placed horizontally atop a pole so that a line drawn from the Omega transmitter to the site makes a right angle with the long axis of the antenna. A button on the receiver activates the LED so the incoming code pattern can be monitored. Irregular flashing means the antenna is incorrectly oriented. When properly oriented, the code consists of two pairs of pulses repeated every ten seconds. Since the code does not distinguish between ten second intervals, the user must provide the minute pulse by releasing the radio reset switch after the new minute. Thus a watch having an accuracy of 5 seconds, or a WWV radio receiver is needed. The reset button is released at the end of the first long pause, of the receiver LED after the minute. The receiver will then automatically synchronize its output to Omega after this. If the button is released more than five seconds after or before the start of the long pause, the clock will run an integral multiple of ten seconds from real time.

The clock must now be set starting with minutes and working up to the Julian day. The receiver automatically takes care of the seconds, so no seconds switch is provided. The delay mentioned in section 3.7 must also be set. This delay is the sum of three components: the receiver delay, propagation delay and atomic time offset from GMT. Table 3 contains pertinent information on the Omega transmitters which can be used to compute the proper settings on the clock. For example, in Anchorage, Alaska the path length from the Hawaii transmitter is 4500km. Multiplying this number by 3.3 x 10**-6 yields a 15ms propagation delay which is added to 4.81s for a total delay of about 4.83ms. Note that the leap seconds often added at the end of June or December cause

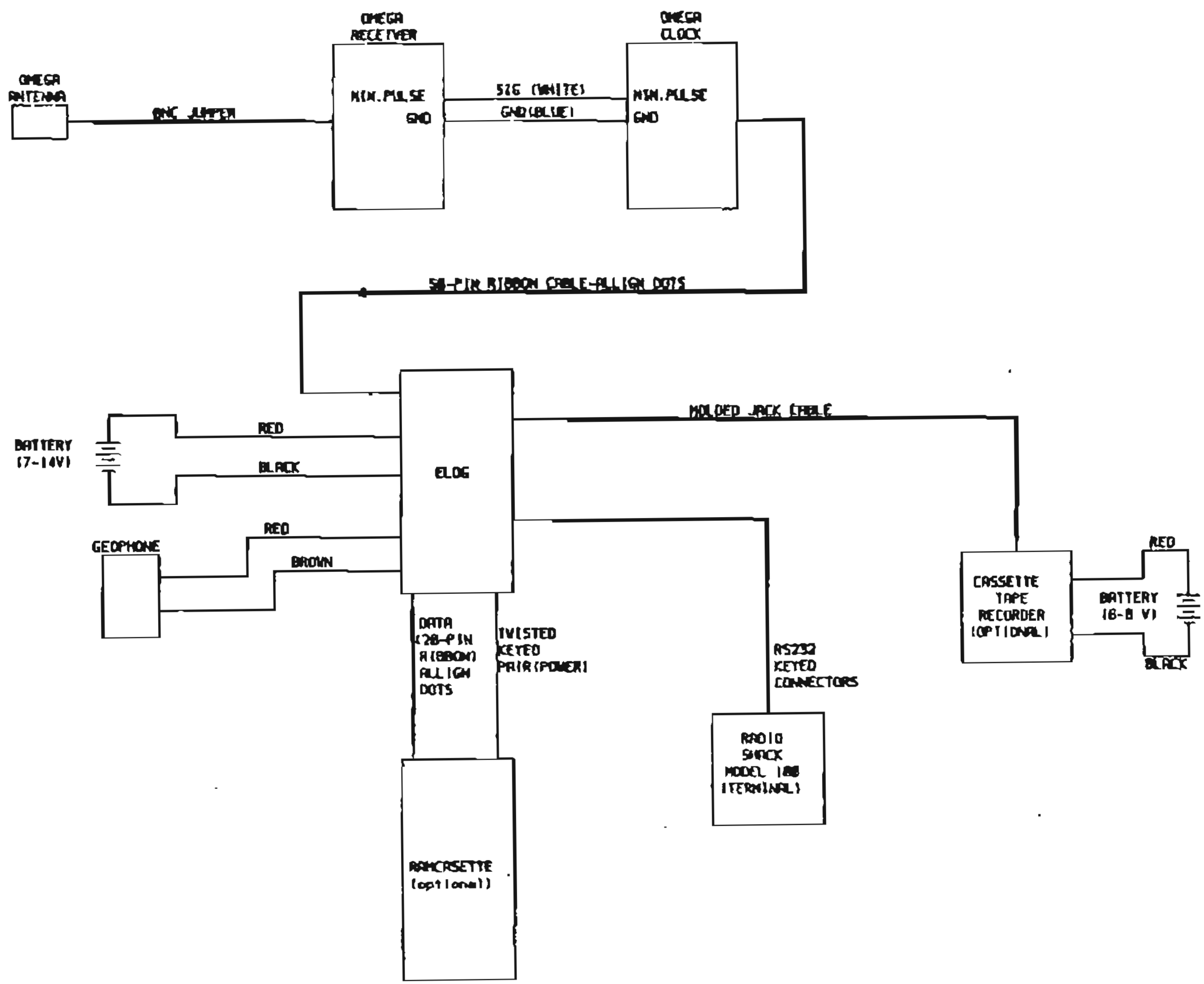


FIGURE 5 - ELOG INTERCONNECTION DIAGRAM

TABLE 3 - OMEGA TRANSMITTER DATA

Station	Offset + Receiver Delay(1/85)
Norway	2.11 sec
Liberia	3.51 sec
Hawaii	4.81 sec
North Dakota	5.91 sec
La Reunion	7.31 sec
Argentina	8.51 sec
Australia	9.61 sec
Japan	0.81 sec

GMT to advance with respect to Atomic time by one second per year. Consult the manufacturer's operation manual [10] for more detailed information. It should also be noted that all Omega transmitters shut down for periodic maintenance. The Hawaii transmitter, for example, is maintained in June. The exact dates can be ascertained by calling the Hawaii Omega Transmitter, 808-235-4981.

Once the delay is set via the three BCD switches on the clock front panel, the clock will approach real time at the rate of 3msec per minute. Thus, if the initial setting were off by 3 seconds, it would take about 17 hours for the time to finally be accurate. During this period, or for any period of poor radio reception, the status bit will be high(logic 1). For good time the status bit will be low (logic 0). It can also be seen that the time needed to "lock" onto real time can be greatly reduced by having a watch accurate to less than +/- 5 seconds.

4.3 SENSOR INSTALLATION

The ELOG front end is set up for the Mark Products L4 geophone. The sensor should be buried as deeply as practical at least 50 feet from the rest of the site. This will help minimize noise coupling from the site to the geophone. A level should also be used to achieve an orientation within +/- 5 degrees to the vertical axis to the geophone axis.

4.4 ELOG setup-step by step

1. Bury the sensor as described above and connect to "GEOPHONE" connector input. Connect red geophone wire to white wire on geophone cable and black to black.
2. Connect the RG-58 coaxial cable from the Omega receiver antenna to "ANT" input.
3. Synchronize the Omega receiver and set the clock as described above.
4. Connect the power cable to "POWER".
5. Connect oscilloscope probe to test points and adjust gain on amplifier until signal is 200-400 mv peak-to-peak. Remove probe.
6. Connect one end of the RS-232 cable to "EIA" connector on CPU board and the other end to the TRS-80 model 100 computer running the terminal emulation program at 1200 baud.
7. Reset CPU switch on Tape I/O board, and type PØ. The ELOG will respond first with an "*" and then after PØ is typed with the MB4TH welcome message. Then type "HEX".
8. Type "XXXX EXECUTE" where XXXX is a hexidecimal address given in Appendix 3 or 4. If cassette tape is being used for storage, depress the rewind button on the tape recorder. When the tape is rewound, press any key and push down the orange record button. If the RAM cassette is being used ignore the rewind commands and depress any key.
9. Verify that the gain setting is correct by tapping the ground several times in quick succession to cause an event to be recorded. (You will also hear the terminal beep.) This should be done at least twice after remaining still for about 2 minutes. If the LTA value sent to the terminal is not in the range of 4-8 adjust the gain switch accordingly.
10. Note the time on the clock, gain setting, LTA value and clock status. The system is now running.

REFERENCES

1. Microboard Development Systems, RCA Solid State Div., Somerville, New Jersey (1982)
2. User Manual for the CDP1802 COSMAC Microprocesor, RCA Solid State Div., Somerville, New Jersey (1976)
3. Microprocessors, Memories, Peripherials, RCA Solid State Div., Somerville, New Jersey (1982)
4. MB4TH, RCA Solid State Div., Somerville, New Jersey (1983)
5. Brody, L., Starting Forth, 1981, Prentice Hall
6. Loelinger, X., Threaded Interpretive Languages
7. McCabe, K. C., Forth Fundamentals, 1983, Dilithium Press
8. User Manual for the RCA COSMAC Microprocesor Development System, RCA Solid State Div., Somerville, New Jersey (1981)
9. Using the CDP1851 Programmable I/O, RCA Solid State Div., Somerville, New Jersey (1982).
10. Omegarec/Omegaface, Observatiore Cantonal, Neuchatel, Switzerland

Appendix 1

Specifications

	ELOG/RAMcassette	ELOG/Cassette Tape
Maximum Input (mV)	7.27/2**GAIN	7.27/2**GAIN
Current Drain (mA)	90	120
Supply Voltage (Volts)	7.5-14	7.5-14
Operating Temperature	-20°C-50°C	0°C-50°C
Mass Storage Capacity*	128 K-bytes (prototype)	80K-bytes
Mass Storage Current Drains	< 1 ma	30 ma
Mass Storage Supply Voltage	5v (from ELOG)	6-7.5 v (ext. battery)
A-D Resolution	8 bits	8 bits
Sampling Rate (SPS)	up to 100	up to 100
Size (with water tight box)	12" x 12" x 22"	12" x 9" x 22"
Amplifier Gain+	340-44,000	340-44,000
Amplifier Noise (at input)	1μV or less	1μV or less
Dynamic Range	48dB	48dB
Desired LTA Value	5-9 counts	5-9 counts
Cost (1984)	2500	2000

* 1/2M-byte RAM cassette has been designed

+ Gain is set digitally by amplifier switch, with lowest value 0 and highest 7

APPENDIX 2 -- INTERRUPT HANDLER

ADDR	CODE	MEMONIC	COMMENT
4688	78	SAV	T->M(R2))
4689	E0	0 SEX	0->X
468A	F8	LDI	
468B	82		operand
468C	A0	0 PLO	82->R(0).0
468D	F8	LDI	
468E	46		operand
468F	B0	0 PHI	46->R(0).1
4690	F8	LDI	
4691	30		operand
4692	50	STR	D->M(R(0)),M(4682)=30
4693	61	1 OUT	30->output,R(1)=4683,sel A-D
4694	F8	LDI	
4695	00		operand
4696	50	STR	D->M(R0)),M(4683)=0
4697	66	6 OUT	0->output,R(1)=4684mfuxed ch.
4698	F8	LDI	
4699	00		operand
469F	50	STR	D->M(R(0),M(4684)=0
469B	65	5 OUT	0->output,R(1)=4686, sel ch.0
469C	F8	LDI	
ADDR	CODE	MEMONIC	COMMENT
469D	09		operand
469E	AB	B PLO	9->R(B).0
469F	2B	B DEC	B-1->B
46A0	8B	B GLO	R(B).0->D
46A1	3A	BNZ	short branch to addr.
46A2	9F	(addr)	branch back to addr
46A3	6B	3 INP	M(4685)=A-D sample byte
46A4	10	0 INC	R(0)=4686
46A5	F8	LDI	
46A6	01		operand
46A7	50	STR	D->M(R(0),M(4686)=1,(semaphore)
46A8	E2	2 SEX	2->X
46A9	68	CID	disable counter interrupt
46AA	00		
46AB	70	RET	return to main program

```
SCR# 0
----- ELOG EARTHQUAKE RECORDER -----
VERSION: 3-14-86
VERSION TYPE: RAMCASSETTE
HIGHEST SCR#: 26
EXECUTION ADDR: 2D0A
SCR# 1
HEX 2000 DP ! ( TM BYTES 4B36-3B)
CODE RDTM
 36 LDI 0 PLO 4B LDI 0 PHI
 8 LDI 0 STR 0 SEX 1 OUT 0 DEC
( SELECT I/O PORT )
 B LDI 0 STR 2 OUT 0 DEC ( A=INP)
53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 9 LDI 0 STR 2 OUT 0 DEC    ( INTB=0)
 1 LDI 0 STR 2 OUT 0 DEC   ( INTA=0)
( READ TIME BYTES)
5 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
4 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
3 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
2 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
1 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
0 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
9 SEP END-CODE
-->
SCR# 2
( CODE DEFINITIONS) HEX
CODE ADPG ( PGRM 1805 CT )
XID CID STPC 49 LDI LDC STM 9 SEP
END-CODE
CODE MIE ( SET MIE, ALLOW INTR )
RET 9 SEP
END-CODE
CODE CTEN ( ENB 1805 CTR INTR )
CIE 9 SEP
END-CODE
CODE CTDS ( DISENB 1805 CTR INTR )
CID 9 SEP
END-CODE
CODE IADR ( INTR HANDLER ADDR )
88 LDI 1 PLO 46 LDI 1 PHI 9 SEP
END-CODE
CODE NRDY (  --  ) ( DATA NOT RDY)
46 LDI 0 PHI 86 LDI 0 PLO 0 LDI
0 STR 9 SEP END-CODE -->
```

```
SCR# 3
( CODE DEFINITIONS) HEX
CODE CTRH ( SET CTR F/F)
 34 LDI 0 PLO 48 LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
98 LDI 0 STR 6 OUT 0 DEC ( F/F=0)
9 SEP END-CODE
CODE CTRL ( CLEAR F/F)
 34 LDI 0 PLO 48 LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
99 LDI 0 STR 6 OUT 0 DEC ( F/F=1)
9 SEP END-CODE
-->
SCR# 4
( CODE DEFINITIONS) HEX
CODE MEMWR ( SET R/W MEM TO WR)
 34 LDI 0 PLO 48 LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 4B LDI 0 STR 2 OUT 0 DEC ( A=OUT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 16 LDI 0 STR 6 OUT 0 DEC ( MEM RD)
 9 SEP END-CODE
CODE MEMRD ( SET R/W = H, READ)
 34 LDI 0 PLO 48 LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 B LDI 0 STR 2 OUT 0 DEC ( A=IN)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 6 LDI 0 STR 6 OUT 0 DEC ( MEM RD)
 9 SEP END-CODE
-->
SCR# 5
( CODE DEFINITIONS) HEX
CODE ENB ( ENABLE MEM BANK SEL'TOR)
 34 LDI 0 PLO 4B LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
96 LDI 0 STR 6 OUT 0 DEC ( ENB DECR)
9 SEP END-CODE
CODE DISB ( DISABLE MEM BANK SEL)
 34 LDI 0 PLO 4B LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
97 LDI 0 STR 6 OUT 0 DEC ( ENB DECR)
9 SEP END-CODE
CODE SBFPTR ( INIT BUF ADDR PTR)
 36 LDI 1 PLO 4B LDI 1 PHI
9 SEP END-CODE
CODE SELPTR ( PTR TO MEM BYTE)
 36 LDI 1 PLO 4B LDI 1 PHI 9 SEP
END-CODE -->
```

```
SCR# 6
( CODE DEFINITIONS) HEX
CODE E->R ( WR FROM ELOG TO RAM)
 34 LDI 0 PLO 4B LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
 4B LDI 0 STR 2 OUT 0 DEC ( A=OUT)
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 1 RNX 1 INC ( LOAD PTR IN RO)
 0 SEX 4 OUT ( WRITE BYTE)
 9 SEP END-CODE
CODE R->E ( WR FROM RAM TO ELOG)
 34 LDI 0 PLO 4B LDI 0 PHI 8 LDI
 0 STR 0 SEX 1 OUT 0 DEC ( SEL PORT)
  B LDI 0 STR 2 OUT 0 DEC ( A=IN )
 53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 1 RNX ( LD PTR TO RO)
0 SEX 4 INP ( READ BYTE)
9 SEP END-CODE
-->

SCR# 7
CODE K/  ( NUM --- REM QUOT ) HEX
PAGEJUMP ( DIV STK NUM BY 4096 )
( RO=DIV'D RF=QUOT RD=REM R1=SGNCTR)
2 SEX 0 LDI 1 PLO IRX ( CLR SGNCTR)
0 RLXA 0 GHI SHL ( LD DIV'D,SGN->DF)
3B C, 15 C, ( BNF POS ADDR...)
1 INC 0 GLO FF XRI 0 PLO 0 GHI
FF XRI 0 PHI 0 INC ( 2'S CML LD SGN)
0 GHI ( ... POS ADDR )
SHR SHR SHR SHR F PLO 0 LDI F PHI
0 GLO D PLO 0 GHI 3 ANI D PHI
1 GLO ( RO=REM, LD SGN CTR)
32 C, 39 C, ( BZ END ADDR... )
F GLO FF XRI F PLO F GHI FF XRI
F PHI F INC ( 2'S COMPL)
D GLO FF XRI D PLO D GHI FF XRI
D PHI D INC ( 2'S COMPL)
2 DEC ( ...END ADDR )
D RSXD F RSXD 9 SEP END-CODE -->
SCR# 8
CODE N/ ( DIVIDE BY 16-NIBBLE)
2 SEX 0 LDI 1 PLO IRX 0 RLXA
0 GHI SHL
3B C, 59 C, ( BNF POS ADDR...)
1 INC 0 GLO FF XRI 0 PLO 0 GHI
FF XRI 0 PHI 0 INC
0 GHI ( ... POS ADDR )
      SHR F PHI 0 GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
0 LDI D PHI 0 GLO F ANI D PLO
1 GLO
32 C, 8D C, ( BZ END ADDR...)
F GLO FF XRI F PLO F GHI FF XRI
F PHI F INC
-->
```

```
SCR# 9
D GLO FF XRI D PLO D GHI FF XRI
D PHI D INC
2 DEC ( ...END ADDR)
D RSXD F RSXD
9 SEP END-CODE
CODE ABV ( NUM -- ABSNUM ) HEX
PAGEJUMP 2 SEX IRX 0 RLXA
0 GHI 80 ANI
32 C, 12 C,
0 GHI FF XRI 0 PHI
0 GLO FF XRI 0 PLO
0 INC 2 DEC 0 RSXD
9 SEP END-CODE
CODE 3* ( NUM -- PROD) ( POS BYTE)
0 SDX ORADCRXSTXD SHBERDC STXD
END-CODE
-->

SCR# 10
CODE YINCF ( ADDR --    ) ( FAST INC)
PAGEJUMP
2 SEX IRX LDXA 1 PHI LDX 1 PLO
1 SEX LDXA 0 PHI LDX 0 PLO
0 INC 0 INC 0 GLO
60 SDI 0 GHI 9 SDBI
33 C, 1B C,
0 LDI STXD STXD
30 C, 1F C,
0 GLO STXD 0 GHI STXD
2 SEX 9 SEP
END-CODE
-->

SCR# 11
( VARIABLES, CONSTS) HEX
VARIABLE RSTA ( REMAINDER FOR STA)
VARIABLE RLTA ( REMAINDER FOR LTA)
VARIABLE PCTR ( TRIG-ONSET DISP)
VARIABLE IMAX ( MAX SAMPLE 1ST )
VARIABLE IMOT ( SAMPLE AT ONSET)
VARIABLE TMXS ( #SAMPLES TO 1ST 0)
VARIABLE OPTR ( PTR FOR ASCII CHAR)
VARIABLE TPTR ( PTR TO HEX TIME)
VARIABLE IPTR ( PTR TO INTR HAND)
VARIABLE YNEW ( NEW SAMPLE)
VARIABLE YOLD ( PREVIOUS SAMPLE)
VARIABLE YCTR ( BUFFER INDEX)
VARIABLE STA ( SHORT TERM AVG)
VARIABLE LTA ( LONG TERM AVG)
VARIABLE TSTA ( TMP LOC FOR STA)
VARIABLE TRSTA ( TMP LOC FOR RSTA)
VARIABLE ECTR ( # TRIGS)
-->
```

```
SCR# 12
( VARIABLES) HEX
VARIABLE N3LT ( # SAMPL < 3*LTA)
VARIABLE NZRO ( # 0-X IN 5 SEC)
VARIABLE TLTA ( TEMP LOC LTA)
VARIABLE TYCTR ( TEMP LOC YCTR)
6000 CONSTANT BOT ( BUF START ADDR)
64 CONSTANT SPS ( SAMPLE RATE)
9 CONSTANT WIND ( TEST WINDOW,SEC)
2 CONSTANT LTRG ( ONSET THRESHOLD)
960 CONSTANT NBUF ( HEX BUFFER SIZE)
-->

SCR# 13
( ELOG WORDS ) HEX
: FAVG YNEW @ ABS STA @ - 4 /MOD
  STA +! RSTA +!
  RSTA @ 4 /MOD STA +! RSTA !
  1 STA @ MAX STA ! ;
: YINC (   --   ) ( INCR YCTR BY 2)
  2 YCTR +! YCTR @ NBUF > IF
  0 YCTR ! THEN ;
: YDEC (   --   ) ( DECR YCTR BY 2)
  -2 YCTR +! YCTR @ 0 < IF
  NBUF YCTR ! THEN ;
: SVBUF (   --   ) ( BUF->RAM)
 CTRL MEMWR ENB SEFPTR
  24 0 DO ( WR 40  BYTES->RAM)
  E->R CTRH CTRL
 LOOP DISB ;
-->

SCR# 14
( ELOG WORDS) HEX
: RDCHAR (   --   ) ( RAM->ELOG)
 CTRL MEMRD ENB SELPTR
 R->E CTRH CTRL DISB ;
: RDREC (   --   ) ( REC->RS232)
  24 0 DO RDCHAR 4B36 C@ . LOOP CR ;
: RDRECS ( N --   ) ( N RECS->RS232)
     0 DO RDREC LOOP ;
-->
```

```
SCR# 15
( ELOG  WORDS ) HEX
: ?RDY (      --      )
  4686 C@ ( WAIT NEW SAMPLE )
  BEGIN DROP 4686 C@ UNTIL SP! ;
: NXT (     --     ) ( REMOVE MEAN )
  YNEW @ YOLD !
  4685 C@ 80 - YNEW ! ; ( 2'S COMPL)
: PUTY YNEW @ YCTR @ BOT  + !
  YCTR YINCF ;
: SAVG   YNEW @ ABV STA @ -  N/
       STA +! RSTA +!
  RSTA @     N/    STA +! RSTA ! ;
-->
```

```
SCR# 16
( ELOG WORDS ) HEX
: LAVG YNEW @ ABV LTA @ -
  K/   LTA +! RLTA +!
  RLTA @      K/   LTA +! RLTA ! ;
: KEEP (  -  ) ( SAVE YCTR,LTA,STA)
  LTA @ TLTA ! STA @ TSTA !
  YCTR @ TYCTR ! RSTA @ TRSTA ! ;
: RSTR (  --  ) ( RESTORE 3 VAR)
  TLTA @ LTA ! TSTA @ STA !
  TYCTR @ YCTR ! TRSTA @ RSTA ! ;
-->
```

```
SCR# 17
( INITIALIZATIONS) HEX
: INIT (  --  ) ( INITIALIZE VAR)
  97 USE ! ( WRITE-->SCR#97)
  8096 4834 ! ( 80=UPDATE,96=SCR#)
  0 YNEW ! 0 YOLD ! 0 YCTR !
  1 STA ! 10 LTA ! 1 TSTA !
  0 RSTA ! 0 RLTA ! 0 PCTR !
  0 IMAX ! 0 IMOT ! 0 TMXS !
  4836 OPTR ! 4836 TPTR !
  4687 IPTR ! 0 N3LT ! F NZRO !
  1 TLTA ! 1 TYCTR !
  0 ECTR ! ;
-->
```

```
SCR# 18
 (. ELOG  WORDS ) HEX
: FRST (    --    ) ( FIRST MOTION)
  1 IMAX ! 1 IMOT ! 0 TMXS ! KEEP
  PCTR @ ABS 0 DO YDEC LOOP
  YCTR @ BOT  + @ IMOT ! ( @ LTRG )
  BEGIN ( LOOK FOR MAX & MIN )
   IMOT @ YCTR @ BOT  + @ XOR 0>
   IMOT @ YCTR @ BOT  + @ = OR
  WHILE
   YCTR @ BOT  + @ ABS IMAX @
   MAX IMAX !
   TMXS 1+! YINC
  REPEAT RSTR ;
-->
SCR# 19
( ELOG WORDS) HEX
: WRBY ( ADDR --  ) DUP
 1 + C@ 2/ 2/ 2/ 2/ OF AND
       OPTR @ C! OPTR 1+!
 1 + C@ OF AND
 OPTR @ C! OPTR 1+! ;
: >> ( BYTE --     ) ( INTR LOADER)
 IPTR 1+! IPTR @ C! ;
: ILD (  --   ) ( INTR:4688-46BE)
 78 >> E0 >> F8 >> 82 >> A0 >>
 F8 >> 46 >> B0 >> F8 >> 30 >>
 50 >> 61 >> F8 >> 00 >> 50 >>
 66 >> FB >> 00 >> 50 >> 65 >>
 F8 >> 09 >> AB >> 2B >> 8B >>
 3A >> 9F >> 6B >> 10 >> FB >>
 01 >> 50 >> E2 >> 68 >> 0D >>
 70 >> ;
-->
SCR# 20
( ELOG  WORDS ) HEX
: WRVR ( ADDR --     ) DUP DUP DUP
  2 0 DO ( WR 4 BYTES-->BUFFER )
   I + C@ 2/ 2/ 2/ 2/ OF AND
        OPTR  @ C! OPTR  1+!
   I + C@ OF AND
   OPTR  @ C! OPTR  1+!
  LOOP ;
: WRTM (     --     )
  6 0 DO ( 11 TIME,1 STATUS -->BUF )
   TPTR @ C@ OF AND
        OPTR  @ C! OPTR  1+!
   TPTR @ C@ 2/ 2/ 2/ 2/ OF AND
        OPTR @ C! OPTR 1+! TPTR 1+!
 LOOP ;
-->
```

```
SCR# 21
( ELOG  WORDS ) HEX
: SRCH (  --  ) ( FIND LOTHRSH TIME)
 0 PCTR ! KEEP ( INITIALIZE)
  NBUF 4 / 0 DO
   YCTR @ BOT  + @ YNEW ! SAVG
   STA @ LTA @ LTRG * <
   STA @ LTA @ LTRG * = OR IF
    LEAVE THEN
   PCTR 1+! YDEC
  LOOP LTA @ STA ! 0 RSTA ! ( REST STA)
40 0 DO ( LOOK BACK FOR START)
 YCTR @ BOT  + @ YNEW ! FAVG
 STA @ LTA @ LTRG * > IF
   LEAVE THEN -1 PCTR +! YINC
LOOP RSTR ; -->

SCR# 22
( ELOG WORDS) HEX
: TAIL (   --   ) ( 5SEC POST EV)
  KEEP ( SAVE YCTR,STA,LTA)
  ADPG CTEN ( START CTR, ENB INTR)
  SPS WIND * 0 DO
   IADR CTEN ?RDY ADPG
   NRDY NXT PUTY LOOP RSTR ;
: AMPL (  --  ) ( DURATION CHECK)
  KEEP 0 N3LT !
  SPS WIND * 0 DO
   YINC
   BOT  YCTR @ + @ YNEW ! SAVG
   STA @ LTA @ 2 * < IF
    N3LT 1+! THEN
  LOOP RSTR ; ( N3LT=#VIOLATIONS)
-->

SCR# 23
( ELOG WORDS ) HEX
: FREQ (  --  ) ( CK #0-X IN 5 SEC)
 0 NZRO ! KEEP ( RESET 0-X CTR)
 SPS WIND * 0 DO YINC
 BOT  YCTR @ + @ ( 1ST POINT)
 BOT  YCTR @ 2 + + @ ( 2ND POINT)
 XOR 0< IF ( SEE IF SIGN CHANGE)
 NZRO 1+! THEN
 LOOP RSTR ;
-->
```

```
SCR# 24
( ELOG WORDS ) HEX
: DOUT (   --   ) ( OUTPUT DATA)
 WRTM YDEC FRST PCTR WRBY IMOT WRVR
 IMAX WRBY TMXS WRBY N3LT WRVR LTA
 WRBY ECTR WRVR NZRO WRVR
 4B36 TPTR ! 4836 OPTR ! SVBUF
  6 0 DO 4B36 I + C@ 0F AND
   2* 2* 2* 2*
   4B36 I + C@ F0 AND 2/ 2/ 2/ 2/ + .
  LOOP CR
." PCTR IMOT IMAX TMXS N3LT LTA ECT
R NZRO" CR SPACE
PCTR ? 2 SPACES IMOT ? 2 SPACES
IMAX ? 3 SPACES TMXS ? 2 SPACES
N3LT ? SPACE LTA ? 4 SPACES
ECTR ? SPACE NZRO ? CR ;
-->
SCR# 25
( ELOG WORDS ) HEX
: ?EVT (   --   ) ( CHECK FOR TRIG)
 STA @ LTA @ 3*   > IF
 RDTM BELL TAIL ( GET TIME FIRST)
 ECTR 1+! ( COUNT EACH BEEP)
 AMPL FREQ SRCH ( VALID. CHECK)
 N3LT @ B4       < ( ENERGY)
 NZRO @ 15        > ( FREQ. TEST)
 PCTR @ 12C       * < ( IMPL. TEST)
 AND AND ( 3 CONDITIONS TRUE?)
 IF DOUT THEN STA @ LTA ! THEN ;
-->
SCR# 26
( ELOG WORDS) HEX
: RUN (   --   ) ( RUN PROG)
 ITAPES INIT ILD CTDS MIE IADR ADPG
BEGIN
 IADR CTEN ?RDY NRDY NXT PUTY SAVG
 IADR CTEN ?RDY NRDY NXT PUTY LAVG
 IADR CTEN ?RDY NRDY NXT PUTY
 1 LTA @ MAX LTA ! ?EVT
0 UNTIL ;
SCR# 27
```

```
 ----- ELOG EARTHQUAKE RECORDER -----
VERSION DATE: 3-14-86
VERSION TYPE: CASSETTE TAPE
HIGHEST SCR#: 21
EXECUTION ADDR: 2BA6
SCR# 1
HEX 2000 DP ! ( TM BYTES 4B36-3B)
CODE RDTM
 36 LDI 0 PLO 4B LDI 0 PHI
 B LDI 0 STR 0 SEX 1 OUT 0 DEC
( SELECT I/O PORT )
 B LDI 0 STR 2 OUT 0 DEC ( A=INP)
53 LDI 0 STR 2 OUT 0 DEC ( B=OUT)
 9 LDI 0 STR 2 OUT 0 DEC    ( INTB=0)
 1 LDI 0 STR 2 OUT 0 DEC   ( INTA=0)
( READ TIME BYTES)
5 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
4 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
3 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
2 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
1 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
0 LDI 0 STR 6 OUT 0 DEC 4 INP 0 INC
9 SEP END-CODE
-->
SCR# 2
( CODE DEFINITIONS) HEX
CODE ADPG ( PGRM 1805 CT )
XID CID STPC 49 LDI LDC STM 9 SEP
END-CODE
CODE MIE ( SET MIE, ALLOW INTR )
RET 9 SEP
END-CODE
CODE CTEN ( ENB 1805 CTR INTR )
CIE 9 SEP
END-CODE
CODE CTDS ( DISENB 1805 CTR INTR )
CID 9 SEP
END-CODE
CODE IADR ( INTR HANDLER ADDR )
88 LDI 1 PLO 46 LDI 1 PHI 9 SEP
END-CODE
CODE NRDY (  --  ) ( DATA NOT RDY)
46 LDI 0 PHI 86 LDI 0 PLO 0 LDI
0 STR 9 SEP END-CODE -->
```

```
SCR# 3
CODE K/   ( NUM --- REM QUOT ) HEX
PAGEJUMP ( DIV STK NUM BY 4096 )
( RO=DIV'D RF=QUOT RD=REM R1=SGNCTR)
2 SEX O LDI 1 PLO IRX ( CLR SGNCTR)
O RLXA O GHI SHL ( LD DIV'D,SGN->DF)
3B C, 15 C, ( BNF POS ADDR...)
1 INC O GLO FF XRI O PLO O GHI
FF XRI O PHI O INC ( 2'S CML LD SGN)
O GHI ( ... POS ADDR )
SHR SHR SHR SHR F PLO O LDI F PHI
O GLO D PLO O GHI 3 ANI D PHI
1 GLO ( RO=REM, LD SGN CTR)
32 C, 39 C, ( BZ END ADDR... )
F GLO FF XRI F PLO F GHI FF XRI
F PHI F INC ( 2'S COMPL)
D GLO FF XRI D PLO D GHI FF XRI
D PHI D INC ( 2'S COMPL)
2 DEC ( ...END ADDR )
D RSXD F RSXD 9 SEP END-CODE -->
SCR# 4
CODE N/ ( DIVIDE BY 16-NIBBLE)
2 SEX O LDI 1 PLO IRX O RLXA
O GHI SHL
3B C, 59 C, ( BNF POS ADDR...)
1 INC O GLO FF XRI O PLO O GHI
FF XRI O PHI O INC
O GHI ( ... POS ADDR )
      SHR F PHI O GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
F GHI SHR F PHI F GLO SHRC F PLO
O LDI D PHI O GLO F ANI D PLO
1 GLO
32 C, 8D C, ( BZ END ADDR...)
F GLO FF XRI F PLO F GHI FF XRI
F PHI F INC
-->
SCR# 5
O D GLO FF XRI D PLO D GHI FF XRI
D PHI D INC
2 DEC ( ...END ADDR)
D RSXD F RSXD
9 SEP END-CODE
CODE ABV ( NUM -- ABSNUM ) HEX
PAGEJUMP 2 SEX IRX O RLXA
O GHI 80 ANI
32 C, 12 C,
O GHI FF XRI O PHI
O GLO FF XRI O PLO
O INC 2 DEC O RSXD
9 SEP END-CODE
CODE 3* ( NUM -- PROD) ( POS BYTE)
2 SEX IRX IRX LDX SHL ADC STXD
O LDI O ADCI STXD 9 SEP
END-CODE
-->
```

```
SCR# 6
CODE YINCF ( ADDR --    ) ( FAST INC)
PAGEJUMP
2 SEX IRX LDXA 1 PHI LDX 1 PLO
1 SEX LDXA 0 PHI LDX 0 PLO
0 INC 0 INC 0 GLO
60 SDI 0 GHI 9 SDBI
33 C, 1B C,
0 LDI STXD STXD
30 C, 1F C,
0 GLO STXD 0 GHI STXD
2 SEX 9 SEP
END-CODE
-->
SCR# 7
( VARIABLES, CONSTS) HEX
VARIABLE RSTA ( REMAINDER FOR STA)
VARIABLE RLTA ( REMAINDER FOR LTA)
VARIABLE PCTR ( TRIG-ONSET DISP)
VARIABLE IMAX ( MAX SAMPLE 1ST )
VARIABLE IMOT ( SAMPLE AT ONSET)
VARIABLE TMXS ( #SAMPLES TO 1ST 0)
VARIABLE OPTR ( PTR FOR ASCII CHAR)
VARIABLE TPTR ( PTR TO HEX TIME)
VARIABLE IPTR ( PTR TO INTR HAND)
VARIABLE YNEW ( NEW SAMPLE)
VARIABLE YOLD ( PREVIOUS SAMPLE)
VARIABLE YCTR ( BUFFER INDEX)
VARIABLE STA ( SHORT TERM AVG)
VARIABLE LTA ( LONG TERM AVG)
VARIABLE TSTA ( TMP LOC FOR STA)
VARIABLE TRSTA ( TMP LOC FOR RSTA)
VARIABLE ECTR ( # TRIGS)
-->
SCR# 8
( VARIABLES) HEX
VARIABLE N3LT ( # SAMPL < 3*LTA)
VARIABLE NZRO ( # 0-X IN 5 SEC)
VARIABLE TLTA ( TEMP LOC LTA)
VARIABLE TYCTR ( TEMP LOC YCTR)
6000 CONSTANT BOT ( BUF START ADDR)
64 CONSTANT SPS ( SAMPLE RATE)
9 CONSTANT WIND ( TEST WINDOW,SEC)
2 CONSTANT LTRG ( ONSET THRESHOLD)
960 CONSTANT NBUF ( HEX BUFFER SIZE)
-->
```

```
SCR# 9
( ELOG WORDS ) HEX
: FAVG YNEW @ ABS STA @ - 4 /MOD
  STA +! RSTA +!
  RSTA @ 4 /MOD STA +! RSTA !
  1 STA @ MAX STA ! ;
: YINC (  --  ) ( INCR YCTR BY 2)
  2 YCTR +! YCTR @ NBUF > IF
  0 YCTR ! THEN ;
: YDEC (  --  ) ( DECR YCTR BY 2)
  -2 YCTR +! YCTR @ 0 < IF
  NBUF YCTR ! THEN ;
-->
```

```
SCR# 10
( ELOG  WORDS ) HEX
: ?RDY (    --    )
  4686 C@ ( WAIT NEW SAMPLE )
  BEGIN DROP 4686 C@ UNTIL SP! ;
: NXT (     --     ) ( REMOVE MEAN )
  YNEW @ YOLD !
  4685 C@ 80 - YNEW ! ; ( 2'S COMPL)
: PUTY YNEW @ YCTR @ BOT  + !
  YCTR YINCF ;
: SAVG   YNEW @ ABV STA @ -  N/
       STA +! RSTA +!
  RSTA @      N/     STA +! RSTA ! ;
-->
```

```
SCR# 11
( ELOG WORDS ) HEX
: LAVG YNEW @ ABV LTA @ -
  K/   LTA +! RLTA +!
  RLTA @       K/   LTA +! RLTA ! ;
: KEEP (  -  ) ( SAVE YCTR,LTA,STA)
  LTA @ TLTA ! STA @ TSTA !
  RSTA @ TRSTA ! YCTR @ TYCTR ! ;
: RSTR (  --  ) ( RESTORE 3 VAR)
  TLTA @ LTA ! TSTA @ STA !
  TRSTA @ RSTA ! TYCTR @ YCTR ! ;
-->
```

```
SCR# 12
( INITIALIZATIONS) HEX
: INIT (  --  ) ( INITIALIZE VAR)
  97 USE ! ( WRITE-->SCR#97)
  8096 4834 ! ( 80=UPDATE,96=SCR#)
  0 YNEW ! 0 YOLD ! 0 YCTR !
  1 STA ! 10 LTA ! 1 TSTA !
  0 RSTA ! 0 RLTA ! 0 PCTR !
  0 IMAX ! 0 IMOT ! 0 TMXS !
  4836 OPTR ! 4836 TPTR !
  4687 IPTR ! 0 N3LT ! F NZRO !
  1 TLTA ! 1 TYCTR !
  0 ECTR ! ;
-->
SCR# 13
( ELOG  WORDS ) HEX
: FRST (   --   ) ( FIRST MOTION)
  1 IMAX ! 1 IMOT ! 0 TMXS ! KEEP
   PCTR @ ABS 0 DO YDEC LOOP
  YCTR @ BOT  + @ IMOT ! ( @ LTRG )
  BEGIN ( LOOK FOR MAX & MIN )
   IMOT @ YCTR @ BOT  + @ XOR 0>
   IMOT @ YCTR @ BOT  + @ = OR
  WHILE
   YCTR @ BOT  + @ ABS IMAX @
   MAX IMAX !
   TMXS 1+! YINC
  REPEAT RSTR ;
-->
SCR# 14
( ELOG WORDS) HEX
: WRBY ( ADDR --  ) DUP
 1 + C@ 2/ 2/ 2/ 2/ OF AND
 30 + OPTR @ C! OPTR 1+!
 1 + C@ OF AND 30 +
 OPTR @ C! OPTR 1+! ;
: >> ( BYTE --    ) ( INTR LOADER)
 IPTR 1+! IPTR @ C! ;
: ILD (  --  ) ( INTR:4688-46BE)
 78 >> E0 >> F8 >> B2 >> A0 >>
 F8 >> 46 >> B0 >> F8 >> 30 >>
 50 >> 61 >> F8 >> 00 >> 50 >>
 66 >> F8 >> 00 >> 50 >> 65 >>
 F8 >> 09 >> AB >> 2B >> 8B >>
 3A >> 9F >> 6B >> 10 >> F8 >>
 01 >> 50 >> E2 >> 68 >> 0D >>
 70 >> ;
-->
```

```
SCR# 15
( ELOG  WORDS ) HEX
: WRVR ( ADDR --     ) DUP DUP DUP
  2 0 DO ( WR 4 BYTES-->BUFFER )
   I + C@ 2/ 2/ 2/ 2/ 0F AND
   30 + OPTR  @ C! OPTR  1+!
   I + C@ 0F AND 30 +
   OPTR  @ C! OPTR  1+!
  LOOP ;
: WRTM (     --     )
  6 0 DO ( 11 TIME,1 STATUS -->BUF )
   TPTR @ C@ 0F AND
   30 + OPTR  @ C! OPTR  1+!
   TPTR @ C@ 2/ 2/ 2/ 2/ 0F AND
   30 + OPTR @ C! OPTR 1+! TPTR 1+!
 LOOP ;
-->
SCR# 16
( ELOG  WORDS ) HEX
: SRCH (   --   ) ( FIND LOTHRSH TIME)
 0 PCTR ! KEEP ( INITIALIZE)
   NBUF 4 / 0 DO
   YCTR @ BOT  + @ YNEW ! SAVG
   STA @ LTA @ LTRG * <
   STA @ LTA @ LTRG * = OR IF
    LEAVE THEN
   PCTR 1+! YDEC
  LOOP
LTA @ STA ! 0 RSTA ! ( RESET STA)
40 0 DO ( LOOK BACK FOR START)
 YCTR @ BOT  + @ YNEW ! FAVG
 STA @ LTA @ LTRG * > IF
   LEAVE THEN -1 PCTR +! YINC
LOOP RSTR ; -->
SCR# 17
( ELOG WORDS) HEX
: TAIL (    --    ) ( 5SEC POST EV)
  KEEP ( SAVE YCTR,STA,LTA)
  ADPG CTEN ( START CTR, ENB INTR)
  SPS WIND * 0 DO
   IADR CTEN ?RDY ADPG
   NRDY NXT PUTY LOOP RSTR ;
: AMPL (   --   ) ( DURATION CHECK)
  KEEP 0 N3LT !
  SPS WIND * 0 DO
   YINC
   BOT  YCTR @ + @ YNEW ! SAVG
   STA @ LTA @ 2 * < IF
    N3LT 1+! THEN
  LOOP RSTR ; ( N3LT=#VIOLATIONS)
-->
```

```
SCR# 18
( ELOG WORDS ) HEX
: FREQ (  --  ) ( CK #0-X IN 5 SEC)
 0 NZRO ! KEEP ( RESET 0-X CTR)
 SPS WIND * 0 DO YINC
 BOT  YCTR @ + @ ( 1ST POINT)
 BOT  YCTR @ 2 + + @ ( 2ND POINT)
 XOR 0< IF ( SEE IF SIGN CHANGE)
 NZRO 1+! THEN
 LOOP RSTR ;
-->
SCR# 19
( ELOG WORDS ) HEX
: DOUT (  --  ) ( OUTPUT DATA)
 WRTM YDEC FRST PCTR WRBY IMOT WRVR
 IMAX WRBY TMXS WRBY N3LT WRVR LTA
 WRBY ECTR WRVR NZRO WRVR
 4 OPTR +! 4B36 TPTR ! OPTR @ 4B2C >
  IF SAVE-BUFFERS 4836 OPTR !
  80 4834 C! 4835 C@ 1 + 4835 C!
  USE 1+! THEN
  6 0 DO 4B36 I + C@ OF AND
   2* 2* 2* 2*
   4B36 I + C@ F0 AND 2/ 2/ 2/ 2/ + .
  LOOP CR
." PCTR IMOT IMAX TMXS N3LT LTA ECT
R NZRO" CR SPACE
PCTR ? 2 SPACES IMOT ? 2 SPACES
IMAX ? 3 SPACES TMXS ? 2 SPACES
N3LT ? SPACE LTA ? 4 SPACES
ECTR ? SPACE NZRO ? CR ; -->
SCR# 20
( ELOG WORDS ) HEX
: ?EVT (  --  ) ( CHECK FOR TRIG)
 STA @ LTA @ 3*  > IF
 RDTM BELL TAIL ( GET TIME FIRST)
 ECTR 1+! ( COUNT EACH BEEP)
 AMPL FREQ SRCH ( VALID. CHECK)
 N3LT @ B4       < ( ENERGY)
 NZRO @ 15        > ( FREQ. TEST)
 PCTR @ 12C       * < ( IMPL. TEST)
 AND AND ( 3 CONDITIONS TRUE?)
 IF DOUT THEN STA @ LTA ! THEN ;
-->
```

```
SCR# 21
( ELOG WORDS) HEX
: RUN (   --   ) ( RUN PROG)
  ITAPES INIT ILD CTDS MIE IADR ADPG
BEGIN
  IADR CTEN ?RDY NRDY NXT PUTY SAVG
  IADR CTEN ?RDY NRDY NXT PUTY LAVG
  IADR CTEN ?RDY NRDY NXT PUTY
  1 LTA @ MAX LTA ! ?EVT
0 UNTIL ;
OK
```

WIRE LIST FOR CLOCK INTERFACE MODIFICATION

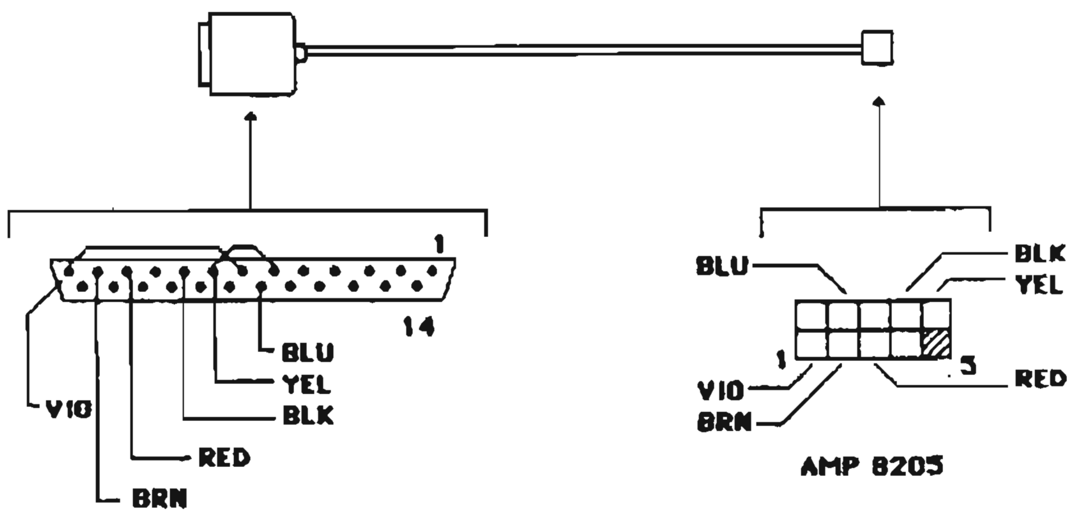
Component and Pin numbers as marked on back of PCB

Male connector pins (Orient w/Batt connections

1. 21-3(1)	26. 21-3(15)
2. 21-2(6)	27. 21-3(14)
3. 21-2(5)	28. 21-3(13)
4. 21-2(4)	29. 21-3(4)
5. 21-2(13)	30. 21-3(5)
6. 21-2(14)	31. 24-4(1)
7. 21-2(15)	32. 24-4(15)
8. 21-2(1)	33. 24-4(14)
9. 21-1(6)	35. 24-4(4)
10. 21-1(6)	36. 24-4(5)
11. 21-1(4)	37. 24-4(6)
12. 21-1(13)	38. 24-4(7)
13. 21-1(14)	39. 21-5(1)
14. 21-1(15)	40. 21-5(15)
15. 21-1(1)	41. 21-5(14)
16. 510-2(2)	
17. 510-2(14)	
18. 510-2(11)	
19. 510-2(6)	
20. 510-1(2)	
21. 510-1(14)	
22. 510-1(11)	
23. 510-1(6)	
24. Batt (+)	
25. Batt (-)	

OUTPUT
CONNECTOR

"Appendix 5A:
Wiring List for Parallel Time Cable"



"Appendix 5B"

ELOG RS232 CABLE

Appendix 6
Automatic Threshold Adjusting

Field experience has shown that the energy threshold, ENTH (see below) is the most critical of the three parameters used to reject uninteresting events. Since the optimun setting for this parameter depends on local noise sources, the memory size available and the duration of the experiment, it is difficult to correctly set this parameter without re-visiting the site at least once.

For example if an experiment were of 10 days' duration and a 2 M-byte RAM cassette were used with waveform storage even a liberal threshold would probably not fill the mass storage as an event every 20 minutes or less would be needed before the storage overflowed. On the other extreme an experiment of one year would definately need to reject the smallest events or the mass storage would overflow before the year elapsed.

With a variable ENTH, the energy threshold used by EVT, this parameter can be automatically increased or decreased depending on the rate at which events are recorded.

Each time the ELOG triggers the time difference between the last time an event was recorded and the time of the present trigger is compared to a desired event recording interval, expressed in minutes. If this time difference is longer than desired, then the threshold is increased and visa versa. This feedback mechanism tends over the long term to force recording at the desired rate so that the mass storage will fill up (but not over-fill) during the experiment.

The words used to implement the above scheme are given below. It is noted that the two undesireable situations which need to be corrected are:

a) ENTH is too high allowing an event recording rate higher than desired.
b) ENTH is too low rejecting so many events that the recording rate is too low.

In the first case ERATE will decrease ENTH by 1 count. In the second case LASTE will increase ENTH by one count. The current value of ENTH will tend to change the recording rate which is reflected in TDIF, the time elasped since the last event recorded. Feedback is accomplished when TDIF is compared to EITVL, the desired recording rate. The words used to implement the above scheme are presented below:

```
:    TDIF ( -- TDIF) ( TDIF LAST EV)
     CVDY DYN ! CVHR HRN ! CVMN MNN !
     DYN   DYO    - 5A0 *
     HRN   HRO    - 3C *
     MNN   MNO    - + + :
:    UPDAT ( -- ) ( UPDATE TEV TIME)
     DYN   DYO ! HRN   HRO !
     MNN   MNO ! :
:    ERATE ( -- ) ( +/- EV RATE)
     TDIF EITVL > if
       ENTH 1 + ! ELSE ENTH 1-! THEN
     ENTH   EMIN MAX ENTH !
```

```
      ENTH   EMAX MIN ENTH ! UPDAT:
:       LASTE ( -- ) ( CK TIME LAST EV )
        TDIF EITVL > IF ENTH 1 + ! THEN
      ENTH   EMIN MAX ENTH !
.     ENTH   EMAX MIN ENTH ! :
:     CVDY ( -- DY ) ( DAY-->BASE 16)
      4B36 C  OF AND 64 *
      4B37 C  OF AND A *
      4B37 C  FO AND 10 / + + :
:     CVHR (  -- HR ) ( HR-->BASE16)
      4B38 C  OF AND A *
      4B38 C  FO AND 10 / + :
:     CVMN ( -- MIN ) ( MIN-->BASE16)
      4B39 C  OF AND A *
      4B39 C  FO AND 10 / + :
:      EVT ( -- ) ( CHECK FOR TRIG )
      STA   LTA   3* > IF
      RDTM BELL TAIL ( GET TIME FIRST )
      ECTR 1 + ! ( COUNT EACH BEEP)
      AMPL FREQ SRCH ( VALID. CHECK)
      N3LT   ENTH   < ( ENERGY TEST)
      NZRO   15 >     ( FREQ. TEST)
      PCTR   32 <     ( IMPL. TEST)
      AND AND ( 3 CONDITIONS TRUE  )
      IF DOUT ERATE ELSE LASTE THEN
      STA   LTA ! THEN :
```